Computing class polynomials with the Chinese Remainder Theorem

Andrew V. Sutherland

Massachusetts Institute of Technology

November 19, 2008

Computing Hilbert class polynomials

Three algorithms

1. Complex analytic
2. p-adic
3. Chinese Remainder Theorem (CRT)

Comparison

Heuristically, all have complexity $O(|D| \log^{3+\epsilon} |D|)$ [BBEL].

Practically, the complex analytic method is much faster ($\approx 50x$)

…and it can use much smaller class polynomials ($\approx 30x$).

Constructing elliptic curves of known order

Using complex multiplication (CM method)

Given p and $t \neq 0$, let $D < 0$ be a discriminant satisfying

$$4p = t^2 - v^2 D.$$

We wish to find an elliptic curve $E/\mathbb{F}_p$ with $N = p + 1 \pm t$ points.

Hilbert class polynomials modulo p

Given a root j of $H_D(x)$ over $\mathbb{F}_p$, let $k = j/(1728 - j)$. The curve

$$y^2 = x^3 + 3kx + 2k$$

has trace $\pm t$ (twist to choose the sign).

Not all curves with trace $\pm t$ necessarily have $H_D(j) = 0$.

Hilbert class polynomials

The Hilbert class polynomial $H_D(x)$

$H_D(x) \in \mathbb{Z}[x]$ is the minimal polynomial of the j-invariant of the complex elliptic curve $\mathbb{C}/\mathcal{O}_D$, where $\mathcal{O}_D$ is the imaginary quadratic order with discriminant D.

$H_D(x)$ modulo a totally split prime ($4p = t^2 - v^2 D$)

The polynomial $H_D(x)$ splits completely over $\mathbb{F}_p$, and its roots are precisely the j-invariants of the elliptic curves E whose endomorphism ring is isomorphic to $\mathcal{O}_D$ ($\mathcal{O}_E = \mathcal{O}_D$).

Practical considerations

We need $|D|$ to be small

Any ordinary elliptic curve can, in principle, be constructed via the CM method. A random curve will have $|D| \approx p$.

We can only handle small $|D|$, say $|D| < 10^{10}$.

Why small $|D|$?

The polynomial $H_D(x)$ is *big*.
We typically need $O(|D| \log |D|)$ bits to represent $H_D(x)$.

If $|D| \approx p$ that might be a lot of bits...

$$H_D(x)$$

Visible
Universe

$\|D\|$	h	$h \lg B$	$\|D\|$	h	$h \lg B$
10^6+3	105	113KB	10^6+20	320	909KB
10^7+3	706	5MB	10^7+4	1648	26MB
10^8+3	1702	33MB	10^8+20	5056	240MB
10^9+3	3680	184MB	10^9+20	12672	2GB
$10^{10}+3$	10538	2GB	$10^{10}+4$	40944	23GB
$10^{11}+3$	31057	16GB	$10^{11}+4$	150192	323GB
$10^{12}+3$	124568	265GB	$10^{12}+4$	569376	5TB
$10^{13}+3$	497056	4TB	$10^{13}+4$	2100400	71TB
$10^{14}+3$	1425472	39TB	$10^{14}+4$	4927264	446TB

Size estimates for $H_D(x)$

$$B = \binom{h}{\lfloor h/2 \rfloor} \exp\left(\pi\sqrt{|D|}\sum_{i=1}^{h}\frac{1}{a_i}\right)$$

Pairing-based cryptography

Pairing-friendly curves

The most desirable curves for pairing-based cryptography have near-prime order and embedding degree k between 6 and 24.

Choosing p and k

We should choose the size of $\mathbb{F}_p$ to balance the difficulty of the discrete logarithm problems in $E/\mathbb{F}_p$ and $\mathbb{F}_{p^k}$. For example

- 80-bit security: $k = 6$ and $170 < \lg p < 192$.
- 110-bit security: $k = 10$ and $220 < \lg p < 256$.

FST, "A taxonomy of pairing-friendly elliptic curves," 2006.

Such curves are very rare...

k	b_0	b_1	10^6	10^7	10^8	10^9	10^{10}	10^{11}	10^{12}	10^{13}
6	170	192	0	0	1	11	33	149	493	1722
10	220	256	0	0	0	0	8	29	85	278

Number of prime-order elliptic curves over $\mathbb{F}_p$ with $b_0 < \lg p < b_1$, embedding degree k, and $|D| < 10^n$.

Karabina and Teske, "On prime-order elliptic curves with embedding degrees $k = 3$, 4, and 6," ANTS VIII (2008).

Freeman, "Constructing pairing-friendly elliptic curves with embedding degree 10," ANTS VII (2006).

Basic CRT method

Step 1: Pick totally split primes

Find $p_1, \ldots, p_n$ of the form $4p_i = t^2 - v^2 D$ with $\prod p_i > B$.

Step 2: Compute $H_D(x) \bmod p_i$

Determine the roots $j_1, \ldots, j_h$ of $H_D(x)$ over $\mathbb{F}_{p_i}$.
Compute $H_D(x) = \prod (x - j_k) \bmod p_i$.

Step 3: Apply the CRT to compute $H_D(x)$

Compute $H_D(x)$ by applying the CRT to each coefficient.
Better, compute $H_D(x) \bmod P$ via the *explicit* CRT [MS 1990].

First proposed by Chao, Nakamura, Sobataka, and Tsujii (1998).
Agashe, Lauter, and Venkatesan (2004) suggested explicit CRT.

Running time of the CRT method

Time complexity

As originally proposed, Step 2 tests every element of $\mathbb{F}_p$ to see if it is the j-invariant of a curve with endomorphism ring $\mathcal{O}_D$.

The total complexity is then $\Omega(|D|^{3/2})$. This is not competitive.

Modified Step 2 [BBEL 2008]

Find a single root of $H_D(x)$ in $\mathbb{F}_p$, then enumerate conjugates via the action of $Cl(D)$, using an isogeny walk.

Improved time complexity

The complexity is now $O(|D|^{1+\epsilon})$. This is potentially competitive. However, preliminary results are disappointing.

Explicit Chinese Remainder Theorem

Standard CRT

Suppose $c \equiv c_i \bmod p_i$, then

$$c \equiv \sum a_i c_i M_i \bmod M,$$

where $M_i = M/p_i$ and $a_i = 1/M_i \bmod p_i$.

Explicit CRT

We can determine $c \bmod P$ directly via

$$c = \left(\sum a_i M_i c_i - rM\right) \bmod P,$$

where r is the closest integer to $\sum a_i c_i / M_i$.

Montgomery and Silverman, 1990.

Space required to compute $H_D(x)$ mod P

Online version of the explicit CRT

The a_i, M_i, and M are *the same* for every coefficient of $H_D(x)$.

These can be precomputed in time (and space) $O(|D|^{1/2+\epsilon})$.

We can *forget* c_i once we incorporate it into running totals for c and r, requiring only $O(\log P)$ bits per coefficient.

Space complexity

The total space is then $O(|D|^{1/2+\epsilon} \log P)$.

This is interesting, but only if the time can be improved.

See Bernstein for other applications of the explicit CRT.

CRT algorithm (split primes)

Given $4P = t^2 - v^2 D$, compute $j(E)$ for all $E/\mathbb{F}_P$ with $\mathcal{O}_E = \mathcal{O}_D$:

1. Construct generating set S for $Cl(D)$.
   Pick totally split primes $p_1, \ldots, p_n$.
   Perform CRT precomputation.
2. For each p_i:
   a. Find $E/\mathbb{F}_{p_i}$ such that $\mathcal{O}_E = \mathcal{O}_D$.
   b. Compute the orbit $j_1, \ldots, j_h$ of $j(E)$ under $\langle S \rangle$.
   c. Compute $H_D(x) = \prod (x - j_k) \bmod p_i$.
   d. Update CRT sums for each coefficient of $H_D(x) \bmod p_i$.
3. Perform CRT postcomputation to obtain $H_D(x) \bmod P$.
4. Find a root of $H_D(x) \bmod P$ and compute its orbit.

Under the GRH: Step 2 is repeated $n = O(|D|^{1/2} \log\log |D|)$ times and every step has complexity $O(|D|^{1/2+\epsilon})$, assuming $\log P = O(\log |D|)$.

Step 2a: Finding a curve with trace $\pm t$

Randomized algorithm

1. Pick E and $\alpha \in E$ until $(p + 1 \pm t)\alpha = 0$.
2. Determine $\#E$ by computing $\lambda(E)$ or $\lambda(\tilde{E})$.
3. If $\#E \neq p + 1 \pm t$ goto Step 1.

Problem

Picking random curves is too slow ($\approx 2\sqrt{p}$ curves to test).

Solution

Don't use random curves!

Generating curves with prescribed torsion

Parameterized families via $X_1(N)$.

For $N \leq 10$ and $N = 12$, parametrizations over $\mathbb{Q}$ [Kubert].

For any N, a point on $X_1(N)/\mathbb{F}_p$ defines a curve $E/\mathbb{F}_p$.

Additional modularity constraints

We can efficiently control $\#E$ mod 3 and $\#E$ mod 4.

Example

Suppose $p + 1 - t$ is divisible by 13 and congruent to 6 mod 12.
We can ensure $\#E \equiv p + t - 1$ mod 132.

Narrows the search by $\approx 110x$ (net speedup $20x$ to $30x$).

See `http://arxiv.org/abs/0811.0296` for details.

Step 2a: Finding a curve with $\mathcal{O}_E = \mathcal{O}_D$

Which curves over $\mathbb{F}_p$ have trace $\pm t$?

There are $H(4p - t^2) = H(-v^2 D)$ distinct j-invariants of curves with trace $\pm t$ over $\mathbb{F}_p$ [Deuring]. For $D < -4$ we have

$$H(-v^2 D) = \sum_{u|v} h(u^2 D).$$

The term $h(u^2 D)$ counts curves with $D(\mathcal{O}_E) = u^2 D$.

What does this tell us?

If $v = 1$ then E has trace $\pm t$ if and only if $\mathcal{O}_E = \mathcal{O}_D$ (easy).
If $v > 1$ then we have $H(4p - t^2) > h(D)$ (harder).

This is a good thing!

Step 1: Pick your primes with care

Problem

There are only $h(D)$ curves over $\mathbb{F}_p$ with $\mathcal{O}_E = \mathcal{O}_D$.
As p grows, they get harder and harder to find: $O(p/h(D))$.
Especially when $h(D)$ is *small*.

Solution [BBEL]

Use a curve with trace $\pm t$ to find a curve with $\mathcal{O}_E = \mathcal{O}_D$ by climbing isogeny volcanoes.

Improvement

We should pick our primes based on the ratio $p/H(4p - t^2)$.
We want $p/H(4p - t^2)$ small. Easy to do when $h(D)$ is big.

Step 2a: Finding a curve with $\mathcal{O}_E = \mathcal{O}_D$

Classical modular polynomials $\Phi_\ell(X, Y)$

There is an ℓ-isogeny between E and E' iff $\Phi_\ell\left(j(E), j(E')\right) = 0$.
To find ℓ-isogenies from E, factor $\Phi_\ell\left(X, j(E)\right)$.

Isogeny volcanoes [Kohel 1996, Fouquet-Morain 2002]

The isogenies of degree ℓ among curves with trace $\pm t$ form a directed graph consisting of a cycle (the surface) with trees of height k rooted at each surface node ($\ell^k \| v$).

For surface nodes, ℓ^2 does not divide $D(\mathcal{O}_E)$.

How to find a curve with $\mathcal{O}_E = \mathcal{O}_D$

Starting from a curve with trace $\pm t$, climb to the surface of every ℓ-volcano for $\ell | v$.

Step 2b: Computing the orbit of $j(E)$

The group action of $Cl(D)$ on $j(E)$

An ideal α in $\mathcal{O}_E \cong End(E)$ defines an ℓ-isogeny

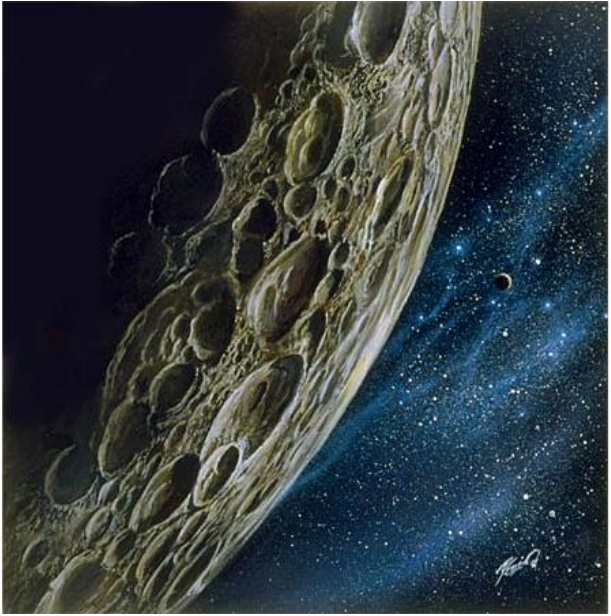
$$E \rightarrow E/E[\alpha] = E',$$

with $\mathcal{O}_{E'} = \mathcal{O}_E$ and $\ell = N(\alpha)$. This gives an action on the set $\{j(E) : \mathcal{O}_E = \mathcal{O}_D\}$ which factors through $Cl(D)$ and reduces mod p for totally split primes (**but ℓ depends on α**).

Touring the rim

We compute this action explicitly by walking along the surface of the volcano of ℓ-isogenies. For $\ell \nmid v$, set $j_1 = j(E)$, pick a root j_2 of $\Phi_\ell(X, j_1)$, then let j_{k+1} be the root of $\Phi_\ell(X, j_k)/(x - j_{k-1})$.

We can handle $\ell | v$, but this is efficient only for very small ℓ.



Step 2b: Computing the orbit of $j(E)$

Walking the entire orbit

Given a basis $\alpha_s, \ldots, \alpha_1$ for $Cl(D) = \langle\alpha_s\rangle \times \cdots \times \langle\alpha_1\rangle$, we compute the orbit of $j = j(E)$ by computing $\beta(j)$ for every $\beta = \alpha_k^{e_k} \cdots \alpha_1^{e_1}$ with $0 \leq e_i < |\alpha_i|$ in a lexicographic ordering of $(e_k, \ldots, e_1)$ (one isogeny per step).

Complexity

Each step involves $O(\ell_i^2)$ operations in $\mathbb{F}_p$, where $\ell_i = N(\alpha_i)$. We need the ℓ_i to be small.

But this may not be possible using a basis!

Representation by a sequence of generators

Cyclic composition series

Let $\alpha_1, \ldots, \alpha_s$ generate a finite group G and suppose

$$G = \langle \alpha_1, \ldots, \alpha_s \rangle \longrightarrow \langle \alpha_1, \ldots, \alpha_{s-1} \rangle \longrightarrow \ldots \longrightarrow \langle \alpha_1 \rangle \longrightarrow 1$$

is a cyclic composition series. Let $n_1 = |\alpha_1|$ and define

$$n_i = |\langle \alpha_1, \ldots, \alpha_i \rangle| / |\langle \alpha_1, \ldots, \alpha_{i-1} \rangle|.$$

Each n_i divides (but need not equal) $|\alpha_i|$, and $\prod n_i = |G|$.

Unique representation

Every $\beta \in G$ can be written uniquely as $\beta = \alpha_1^{e_1} \cdots \alpha_s^{e_s}$, with $0 \le e_i < n_i$ (we may omit α_i for which $n_i = 1$).

Step 1: Generating system for $Cl(D)$

A generating set for $Cl(D)$

Represent $Cl(D)$ with binary quadratic forms $ax^2 + bxy + cy^2$.
Under GRH, forms with prime $a \leq 6\log^2|D|$ generate $Cl(D)$.

Norm-minimal generating system S

Let $\alpha_1, \ldots, \alpha_s$ be the sequence of primeforms ordered by a.
Let S be the subsequence of α_i with $n_i > 1$.

Computing the n_i

We can compute the n_i using either $O(|G|)$ or $O(|G|^{1/2+\epsilon}|S|)$ group operations with a generic group algorithm.

A back-of-the-envelope complexity discussion

Some useful facts and heuristics

1. $h(D) \approx 0.28|D|^{1/2}$ on average.
2. $\max p_i = O(|D| \log^{1+\epsilon} |D|)$ heuristically ($p_i \ll 2^{64}$).
3. $\max \ell = O(\log^{1+\epsilon} |D|)$ conjecturally, and for most D, $\max \ell = O(\log \log |D|)$ heuristically.

Which step is asymptotically dominant?

If $\mathbb{F}_{p_i}$ adds/mults cost $O(1)$, for most D we expect:

1. Step 2a has complexity $O(|D|^{1/2} \log^{1.5+\epsilon} |D|)$.
2. Step 2b has complexity $O(|D|^{1/2} \log^{1+\epsilon} |D|)$.
3. Step 2c has complexity $O(|D|^{1/2} \log^{2+\epsilon} |D|)$.

For exceptionally bad D, Step 2b is $\Omega(|D|^{1/2} \log^2 |D|)$.

Step 2c: Computing $H_D(x) = \prod(x - j_k) \bmod p_i$

Building a polynomial from its roots

Standard problem with a simple solution: build a product tree.
Using *FFT*, complexity is $O(h \log^2 h)$ operations in $\mathbb{F}_{p_i}$.

Harvey's experimental znpoly library

Fast polynomial multiplication in $\mathbb{Z}/n\mathbb{Z}$ for $n < 2^{64}$, via multi-point Kronecker substitution. Two to three times faster than NTL for polynomials of degree 10^3 to 10^6.

`http://cims.nyu.edu/~harvey/`

$-D$	$12,901,800,539$	$13,977,210,083$	$17,237,858,107$
$h(D)$	54,076	20,944	14,064
$\lceil \lg B \rceil$	5,497,124	2,520,162	1,737,687
ℓ_1	3	3	11
ℓ_2	5		23
$Cl(D)$ time	0.1	0.3	0.2
n	141,155	68,646	47,302
$\lceil \lg(\max p_i) \rceil$	42	39	38
prime time	3.9	1.3	1.9
CRT pre time	2.8	0.9	0.6
CRT post time	0.9	0.9	0.6
(a,b,c) splits	**(56,14,30)**	**(81,7,13)**	**(50,48,2)**
Step 2 time	**70,600**	**27,000**	**45,300**
root time	347	171	67
roots time	220	132	130

CRT method computing $H_D \bmod P$ (MNT curves, $k = 6$)

(2.8GHz AMD Athlon CPU times in seconds)

Class invariants and class polynomials

The j-invariant $j(\tau)$

For $\tau \in \mathbb{H}$, define $j(\tau) = j(E_\tau)$, where $E_\tau = \mathbb{C}/[1, \tau]$.

1. $\mathbb{Q}(j(\tau))$ is the ring class field of $\mathcal{O}_D \cong End(E_\tau)$.
2. The min. poly. of $j(\tau)$ is $\mathcal{P}_j(x) = H_D(x)$ (for any τ).

Other class invariants $\varphi(\tau)$

If $\mathbb{Q}(\varphi(\tau)) = \mathbb{Q}(j(\tau))$, we call $\varphi(\tau)$ a *class invariant* [Weber].

We want φ to satisfy (2) (not always true) and to have an algebraic relationship with j.

$\mathcal{P}_\varphi(x)$ may have *much* smaller coefficients than $H_D(x)$.

Alternative class invariants [with Enge]

A simple example (assume $3 \nmid D$)

The function $\gamma_2 = \sqrt[3]{j}$ is a class invariant satisfying (2).

A minimally modified algorithm:

1. Reduce height estimate by a factor of 3.
2. Restrict to $p_i \equiv 2 \bmod 3$ so that cube roots are unique.
3. Compute $\gamma_2 = \sqrt[3]{j}$ for each j enumerated in Step 2b.
4. Form $\mathcal{P}_{\gamma_2}(x) = \prod(x - \gamma_2)$ instead of $H_D(x)$ in Step 2c.
5. Cube a root of $\mathcal{P}_{\gamma_2}(x) \bmod P$ to get desired j at the end.

Variations

- It is also possible to use $p_i \equiv 1 \bmod 3$ [Bröker].
- One can enumerate γ_2 directly in Step 2b.

Better class invariants for the CRT method

For $3 \nmid D$ and $|D| \equiv 7 \bmod 8$ use f^2 [Weber]

Use $p_i \equiv 11 \bmod 12$ to determine f^2 over $\mathbb{F}_{p_i}$ via

$$\gamma_2 = (f^{24} - 16)/f^8.$$

Reduces the height bound by a factor of 36.

For $|D| \equiv 11 \bmod 24$ use g^2 [Ramanujan]

Use $p_i \equiv 2 \bmod 3$ to determine g^2 over $\mathbb{F}_{p_i}$ via

$$\gamma_2 = g^6 - 27g^{-6} - 6.$$

Reduces the height bound by a factor of 18.

When constructing an elliptic curve of prime order, we have $|D| \equiv 3 \bmod 8$.

	j	γ_2	g^2
$\lceil \lg B \rceil$	5,497,124	1,832,376	305,397
n	144,145	49,097	8,768
splits	(56,14,30)	(42,22,36)	(18,42,40)
Step 2 time	70,600	19,600	2,940
speed up	-	3.6	24

CRT method class invariant comparison

$D = -12,901,800,539 \qquad h(D) = 54,076$

		Complex Analytic		CRT Method		
$-D$	$h(D)$	bits	time	bits	time	**ratio**
6961631	5000	9.5k	28	7.5k	7	**4**
23512271	10000	20k	210	16k	29	**7**
98016239	20000	45k	1,800	35k	140	**13**
357116231	40000	97k	14,000	76k	650	**22**
2093236031	100000	265k	260,000	207k	4,600	**57**

Complex Analytic (double η quotient) vs.
CRT method (f^2)
(2.4 GHz AMD Opteron CPU seconds)

Enge, "The complexity of class polynomial computations via floating point approximations" (2008)

Scalability

Distributed computation

Elapsed times on 14 PCs run in parallel (2 cores each):

$D = -10,149,832,121,843,$	$h = 690,706$	**11 hours**
$D = -102,197,306,669,747,$	$h = 2,014,236$	**4.6 days**

Using Ramanujan invariant g^2.

Minimal space requirements

Under 300MB memory (per core). Total storage under 2GB. (Class polynomial over $\mathbb{Z}[x]$ is more than 4TB.)

Plenty of headroom

Larger computations are feasible.

$-D$	$h(D)$	bits	primes	time	split
$10^6 + 19$	342	1.3k	65	¡0.1	(43,50,7)
$10^7 + 19$	1,140	5.2k	222	1.0	(24,61,15)
$10^8 + 19$	3,258	16k	597	8.2	(35,49,16)
$10^9 + 19$	10,478	57k	1,909	110	(28,42,30)
$10^{10} + 19$	39,809	220k	6,561	1,700	(21,38,41)
$10^{11} + 19$	160,731	970k	25,431	34,000	(14,34,52)
$10^{12} + 19$	366,468	2.6m	63,335	230,000	(21,30,50)
$10^{13} + 19$	1,360,096	10m	223,637	3,600,000	(15,27,58)
$10^{14} + 43$	2,959,552	25m	523,719	22,000,000	(20,25,55)

CRT method using Ramanujan invariant ($|D| = 11 \bmod 24$)

(Estimated 2.8 GHz AMD Athlon CPU seconds)